1. Solution explanation:

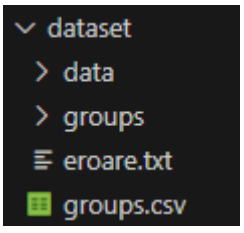
The solution follows a structured approach to classify and group website logos efficiently:

- **Data Conversion**
  - I started by converting the provided .parquet dataset into a .csv file to make it easier to extract website domains.
- **Logo Extraction**
  - I extracted domain names from the CSV and downloaded their favicons using Google's Favicon API.
  - If a logo couldn't be retrieved, I logged the domain in an error file for further review.
- **Logo Processing & Clustering**
  - Duplicate Removal: I used Perceptual Hashing (pHash) to detect and eliminate near-identical logos.
  - Feature Extraction: I computed Hu Moments to capture shape-based characteristics for each unique logo.
  - Clustering: I applied DBSCAN to group similar logos based on extracted features.
  - Output Generation: I saved logos in organized cluster folders and generated a CSV file mapping each domain to its respective group.

2. Output:

The program processes a set of website logos and outputs multiple clusters, grouping visually similar logos together.

- **Initial Image Count**
  - The program starts with 3162 logos.
- **Duplicate Removal**
  - Using **Perceptual Hashing** - identifies and removes duplicate images, leaving 1168 unique logos.
- **Final Output**
  - The clusters are saved in separate folders, each containing similar logos.
  - A CSV file is generated, mapping each domain to its cluster.
  - The cluster distribution shows that some groups have a high number of similar logos (e.g., cluster -1 with 531 images), while others contain only a few.



3. **Code and Logic:**

- convert_parquet_to_csv.py: I used this code to convert the .parquet file to .csv to make data extraction easier.
- extract_logos.py: The code reads a list of domain names from logos.csv, then downloads their favicons from Google. The images are saved in a designated folder, while any failed downloads are logged in an error file. At the end, it prints a completion message.
- cod.py: The code uploads logo images, removes duplicates using pHash, which detects visual similarities even if the images are slightly modified. It then extracts features based on the logo shape using Hu moments, useful for visual recognition. For clustering, it uses DBSCAN because it does not require a fixed number of clusters and can identify outliers, unlike K-Means or KNN. Finally, the images are organized in folders and the results are saved in a CSV.